On Open Source IR

A. MacFarlane,
Centre for Interactive Systems Research, City University, London EC1V 0HB

Abstract: Open source software development is becoming increasingly popular as a way of producing software, due to a number of factors. It is argued in this paper that these factors may have a significant impact on the future of IR systems, and that it is desirable that these systems are made open to all. We outline some problems that may prevent the uptake of open source IR systems. A number of open source IR systems are described.

1.Introduction

I argue here that open source software development is a promising method for producing Information Retrieval (IR) systems. Open source software is available freely (usually on the Web), and the source code is available so that changes may be made as required. Software which enjoys wide use has the most to gain from open source, and IR systems are likely to be increasingly used as more and more people and the organisations they work for require Web or intranet search. The paper is organised a follows. In order to give the reader some background in the area of open source software, we give a description of this type of development in section 2. In section 3 we outline some of the systems that are mature and available for use by IR researchers and practitioners. In section 4 we provide an argument for Open Source IR systems, and outline the potential problems and obstacles that may counter the benefits of Open Source in IR in section 5. An agenda is set for Open Source IR software in a conclusion.

2. Open source software

Open source software (Feller & Fitzgerald, 2002) is where the source code of programs is made freely available for anyone to change and distribute providing they abide by the accompanying licence. This differs from closed source or propriety software which may only be obtained by some form of payment either by purchase or by 'leasing'. The difference between open and closed source can be characterised by the word *freedom*: users of open source software have the freedom to alter the source code while users of closed source software do not.

There are many licenses available for the use with open source software (Feller & Fitzgerald, 2002), but the most popular of these is the GNU Public Licence (GPL). The GPL gives programmers the right to alter software and re-distribute it, providing that the changes they have made are available to other programmers: this concept is known as *copyleft*. Thus the GPL is said to be 'viral' (Feller & Fitzgerald, 2002): any software which uses the GPL must itself be released under the GPL. The reasoning behind copyleft is to prevent propriety software houses from exploiting the (often unpaid) work of open source programmers, without some recompense.

3.Systems currently available

There is a wide range of Open Source IR systems available on the Web from sites such as sourceforge.net and freshmeat.net. These systems can be categorised into two main groups: those which use inverted files and those which use databases systems. We provide a short description of small number of systems, which are mature or have been used by either IR practitioners or researchers and which used the inverted file approach. All of these systems are free software and most are released under the GNU Public Licence (GPL). The systems we concentrate on are Xapian, Swish++, Senga, ht://Dig, Isearch Oasis, MG and Lemur (other freely available IR systems found in the course of our investigation are listed in appendix 1).

3.1 Xapian (http://sourceforge.net/projects/xapian/)

Xapian has had something of a chequered history. It was originally developed under the name of Open Muscat (Porter & Boulton, 2000) by BrightStation PLC, changed its name to Omsee and then to OmSeek when the BrightStation open source project was closed down (due to financial problems unrelated to the project itself), and then to Xapian. However, it continues to be developed under the GNU Public Licence (GPL).

Xapian is not a program but a suite of libraries providing an Application Programming Interface (API) for services such as indexing, search and relevance feedback. Stemming functions are provided for English and many other European languages. The library is written in C++, but API's to other languages such as Perl and Python are also available. The model provided as part of search is the Robertson/Sparck Jones probabilistic model (Robertson & Sparck Jones, 1976) and the weighting function used is BM25 (Robertson et al, 1995). Boolean and phrase search facilities are also provided. Examples of how to use the search and index facilities are provided.

3.2 Swish++ (http://homepage.mac.com/pauljlucas/software/swish/)

Swish++ (**S**imple **W**eb **I**ndexing **S**ystem for **H**umans) is a Unix based indexing and search engine that has been ported to MS Windows. It is a C++ rewrite of Swish-E (http://sunsite.berkeley.edu/SWISH-E/) a system developed for Web search. Swish++ offers command line services for Web crawling, indexing and facilities to create a search server. Non-text documents such as Microsoft Word documents can be indexed. Swish++ at present only handles the English language. The search model used in the software is Boolean, but some form of ranking function is provided (the term weighting model is not specified). A significant problem with both systems is that neither Swish-E nor Swish++ has the ability to merge intermediate results: the user is expected to increase the memory capacity of the hardware in order to handle large collections. This is not an appropriate strategy for an information retrieval system.

3.3 Senga (http://www.senga.org/)

Senga (like Xapian) is a library of C++ routines for developing information retrieval systems. Unlike Swish++ the developer must write their own query and document

parsers. No stemming facilities are provided unlike Xapian. Senga therefore provides a basic library for IR systems development. Senga shares some programming effort with ht://Dig, a good example of collaboration between two development groups. The system appears to allow programmers to define their own weighting function, so in theory any term weighting model could be supported. Senga has the ability to support updates, therefore dynamic indexes can be supported. The author had problems in building executables from source code for this system: no answer was ever provided to my query and development work appears to have stalled.

3.4 ht://Dig (http://www.htdig.org/)

The ht://Dig system has already been mentioned in connection with Senga. This system provides programs to crawl Web sites (or *Dig* them in their nomenclature) merge data from existing database or newly found documents, and finally to search document and word databases. Search facilities include ranking and Boolean models, as well as special functions for multiple keyword search and building synonym databases. The ranking function is an ad-hoc one and scores words nearer the top of the document higher than those nearer the bottom. The ability to define a front-end web page for search is provided. The system appears to be quite widely used and there is much development work being done on it. There appears to be some problems on Linux versions: the author wasted the best part of a day trying to fix an intermittent bug that crashed the system and was impossible to find.

3.5 Isearch (www.etymon.com/Isearch)

Isearch was written to resolve perceived problems with freeWAIS, in particular that the search engine and retrieval protocol were mixed in together (Nassar, 1997). Apart from the article by Nassar (1997) there is very little documentation on the detailed facilities, but looking at the source code modules it is clear that functions such a geographical and date search are supported. Ranking is supported, but the model and functions used by the system is not declared. Nested Boolean searches are allowed as is phrase searching and right truncation. Isearch gives the user the ability to index various formats such as HTML, SGML, Medline and USMARC. The code is not longer being developed and a replacement called Amberfish® (which is able to deal with structured XML documents) has been developed to replace it.

3.6 Oasis (http://oasis-europe.org)

Oasis (Kluev, 2000) differs from the systems described above, as it is a program for distributed search on the Internet. The base system used for search is Isearch (see section 3.5). It filters out collections using server selection methods and submits queries to distributed sites, merging the results from them. Oasis uses artificial intelligence and neural network techniques for server selection and query processing. The architecture used is basically three-tier middleware: the client sends a query to any Oasis server which contacts other Oasis servers available to obtain results. Oasis provides synchronous and asynchronous modes: in the latter the server acts as an intelligent agent. A collection

broker chooses the servers, but it is not clear how server selection is done. Results are merged using a Neural Network technique that also removes duplicate entries: it is entirely possible that duplicate sets of results can be generated in this type of architecture. Crawler services are provided for collection building: these are described in Kluev (2000). The system also supports relevance feedback.

3.7 MG (http://www.cs.mu.oz.au/mg/)

Managing Gigabytes or MG for short (Witten, Moffat and Bell, 1999) is a text retrieval system which has been developed to investigate various aspects of compression for inverted lists. The code is released under the GPL, but is not longer being actively developed: further development is in the closed source model. The system provides facilities for indexing collections and searching them using either the Boolean or Vector Space models. The book by Witten, Moffat and Bell (1999) gives detailed information on the compression methods used in MG: there is also guide to MG in the appendices of the book.

3.8 Lemur (http://www-2.cs.cmu.edu/~lemur)

Lemur is a toolkit which is being actively developed by both the University of Massachusetts and Carnegie Mellon University to be used for research into the areas of language modelling and information retrieval. The toolkit allows programmers to develop all kinds of information retrieval systems such as cross-lingual IR and text filtering/classification. The system can either work on Unix or Windows systems. The license used is a private one.

4.The argument for open source IR systems

A number of factors demonstrate that open source software development works well (Moody, 2001). The biggest factor is parallel development and debugging. In closed systems development, programming effort is restricted to a small team who are the only people who can address problems in the code. It is difficult for team members to test their software against real problems. With open source software we may have programming effort shared amongst programmers who may be widely distributed all over the world. The number of programmers who can tackle development is far more than any organisation that uses closed source development could ever deploy. Having this wide resource of programmers allows open source developers to distribute the functions of debugging and testing, deploying systems to real problems and identifying problems and errors which can be tackled far quicker than in closed source development. It is estimated that up 70% to 80% of the cost of software is on maintenance (P147: Feller and Fitzgerald): the potential economic benefits of sharing these overheads are clear. Another clear advantage is in the management of the projects. Although there will be one or two project leaders in open source projects, in the main each developer will work independently without the need for any significant direction (they volunteer to tackle a well defined specific problem) or the need for communication with everyone on the team. In closed source development, significant project management is required which results

in problems outlined by Brooks (1982), notably the offset between using more programmers on a project and the level of communication between them.

The significance of this type of development is that new IR algorithms and models can be released into the community very quickly for practical application and deployment. This contrasts with IR ideas developed in closed source systems. These remain secret and unused, often untested and uncompared using standard test collections such as TREC (Voorhees and Harman, 1999). These ideas are lost to the community who have no chance to discuss their impact. Porter and Boulton (2000) assert that mutual suspicion between IR practitioners in industry and IR researchers in academia was caused by closed and proprietary software development methods. To the industrialists, IR researchers did not understand the realities of dealing with real world problems and were stuck in theoretical ideas that do not tackle them. To the academics, people in industry did not understand the theory of IR and who stole their ideas without giving them the credit they thought was due. Porter and Boulton (2000) argue that this lack of co-operation between the two camps has been damaging to both.

While the survey in section on 'Systems currently available' is by no means exhaustive, it is clear that some of the arguments given here are illustrated by examples in open source systems. There is clear co-operation between Senga and ht://Dig as well as between Oasis and Isearch, for the benefit of both sets of groups. However it is also clear that some open source developers are completely unaware of progress made in IR theory: term weighting functions used in many surveyed systems are either undeclared or ad-hoc with little or no theoretical underpinning to them. It is very hard for software developers, who are interested in developing systems, to keep up with new ideas emanating from the world of IR research.

5. Potential problems and obstacles

Although open source development has shown in many cases to be successful, there are potential problems (Moody, 2001). Commercial companies do not always see the benefits, and are under significant pressure from shareholders who do not understand the open source development model. A case in point is BrightStation and Xapian. This (and other human factors) may cause *forks*. A fork occurs in open source development when a group developing open source software splits into two or more groups. This means that the groups will often duplicate work, wasting time and effort obviating the benefits listed above for open source development.

The impact on IR system developers can be considerable. The author found 36 open source IR systems in the course of his investigation. It is entirely possible that there are more systems available on the Web and from other sources. Proliferation may be no bad thing to begin with in order to share ideas, but at some stage it would be a good idea to coalesce the ideas from these systems in order to get the best of them and hence to share them. In the long run it is desirable that only a few systems be available for reasons given above. However the author does not want to put off IR researchers or developers from releasing their code: going open source is another way of sharing ideas. One of the main reasons for publishing the material here is to encourage IR practitioners and researchers to share ideas, coalesce them from different systems and to prevent forks before they happen. The author accepts that it will be impossible to coalesce all the open

source IR systems into one for significant political, social and software engineering reasons (e.g. differences in programming languages).

There is potential for wasted effort in finding problems in the software. For example, we may have many programmers looking at the same chunk of code, duplicating debugging effort and therefore reducing overall efficiency of the open source software development model (McConnell, 1999). There is also a workload deployment issue: for example some programmers have stronger expertise in some areas than other programmers (McConnell, 1999). There is an offset when maintaining software in the benefits of sharing maintenance and loss due to wasted effort. This is a significant area of research in open source software (Feller and Fitzgerald, 2002).

Another significant factor and one that has received little attention is usability. Nichols et al (2001) argue that open source software projects need to adapt in order to produce systems that can be used by a typical and non-technical user. The basic problem is that most open source systems are written by programmers who do not understand end user needs and whose software is often complex and difficult to use. For example, a quick look at some of the indexing methods use by systems in this survey has revealed how difficult and complicated creating an index is. Often significant technical knowledge is needed in order to make the process work correctly or efficiently. This is a serious problem, and given the importance of interaction in IR systems it is one that must be addressed by open source IR developers.

6.Conclusion: the way forward

The benefits of sharing ideas between IR industrialists and researchers are clear. People in industry often have access to users which IR researchers can only dream of. Ideas can be tested out in real situations, often anonymously without reference to the users or publishing the material. These ideas can be embodied in the systems themselves and downloaded for use in either research or deployment in real world search. This author believes that good new ideas should be available to all, and both IR industrialists and academics stand to gain from closer co-operation. I want to encourage programmers who work on open source IR systems to consider the issue of usability and where possible to coalesce their systems. The author is aware however that efforts are being dissipated in the development of open source IR systems, and may this prevent the obvious benefits in using the open source development model. I hope that further research into the open source development model will be able to show tangle benefits, and that this will impact positively on using such a model to produce IR systems.

Acknowledgements

I am grateful to Francois Schiettecatte, Martin Porter and Richard Boulton for their comments on an early draft of this paper.

References

Brooks, F.P. (1982). The mythical man-month: essays on software engineering, Addison-Wesley, Reading: Massachusetts.

Feller, J, and Fitzgerald, B. (2002). Understanding Open Source software development, Addison-Wesley, London.

Kluev, V. (2000). Compiling document collections from the internet, *SIGIR Forum*, 34(2), 9-14.

Nassar, N. (1997). Searching with Isearch.
URL: http://www.webtechniques.com/archives/1997/05/nassar/ (visited 9th December 2002).

Nichols, D.M, Thomson, K and Yeates, S.A. (2001). Usability and open source Software development. Department of Computer Science, University of Waikato.
URL: http://www.cs.waikato.ac.nz/~say1/pubs/oss.pdf (visited 15th January 2003).

McConnell, S. (1999). Open source methodology: ready for prime time? IEEE Software 16(4), July/August 1999, 6-11.

Moody, G. (2001). Rebel code: Linux and the open source revolution, Allen Lane, London.

Porter, M.F. & Boulton, M. (2000). Open Muscat, an Open Source search engine, *SIGIR Forum*, 34(1), 16-17.

Robertson, S.E. & Sparck Jones, K. (1976). Relevance weighting of search terms, *Journal of the American Society of Information Science*, May-June, 129-145.

Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., and Gatford, M. (1995). Okapi at TREC3, In: Harman, D.K. (ed). *Proceedings of Third Text Retrieval Conference, Gaithersburg, USA, November 1994, NIST SP 500-226*, (Gaithersburg: NIST), 109-126.

Voorhees, E.M. & Harman, D.K. (eds). (1999).The Seventh Text Retrieval Conference (TREC-7), NIST Special Publication 500-242, NIST.

Witten, I.H., Moffat, A., and Bell, T.C. (1999). Managing Gigabytes (2nd Edtion), Morgan Kaufmann, San Francisco: California.

Appendix 1 – Other Open Source/Free Software IR projects

Project Name	URL	Stage	Licence	Other Information
Alkaline	alkaline.vestris.com	Stable	Private	Free for non-commercial use.
Amberfish®	www.etymon.com/amberfish	Stable	GPL	Process structured XML.
Glimpse	webglimpse.net	Stable	Private	Free for non-profit organisations.
Ongobongo	ports.tolkien.dk/ongobongo	Alpha	GPL	Uses PostgresQL.
Perlfect	perlfect.com/freescripts/search/	Stable	GPL	-
ROADS	www.roads.lut.ac.uk	Stable	GPL	Yahoo like.
SEARCH.PHP3	www.w3webmaster.com/search/install.shtml	Stable	GPL	Uses MySQL.
YASE	www.mazumdar.demon.co.uk/yase_index.html	Beta	GPL	-
Exist	exist.sourceforge.net	Beta	LGPL	Uses RDMS
PhpDig	phpdig.toiletoine.net	Stable	GPL	Uses MySQL
NISs	sourceforge.net/projects/niss	Planning	GPL	-
Anarchivist	sourceforge.net/projects/anarchivist	Planning	GPL	Rewrite of AustLII
ASPSeek	www.aspseek.org	Stable	GPL	Uses SQL
HISS	sourceforge.net/projects/hiss	Planning	GPL	-
Latente	sourceforge.net/projects/latente/	-	-	Uses Java.
NeatSeeker	neatseeker.sourceforge.net	Stable	Apache	-
Nose	sourceforge.net/projects/nose	Planning	GPL	-
OpenFTS	openfts.sourceforge.net	Stable	GPL	Uses PostgresQL.
Oxyus	sourceforge.net/projects/oxyus	Pre-Alpha	Apache	-
Ransacker	ransacker.sourceforge.net	Alpha	GPL	Uses Python
siteIndexer	sourceforge.net/projects/siteindexer	Alpha	GPL	Uses MySQL
SPINdex	mattwork.potsdam.edu/projects/spandex	Stable	GPL	-
Doc Fox	sourceforge.net/projects/docfox	Planning	BSD	-
MPS Information Server	www.fsconsult.com	Stable	Private	-
Lucene	jakarta.apache.org/lucene/	Stable	Apache	Uses Java.
freeWAIS-sf	Is6-www.informatik.uni-dortmund.de/ Ir/projects/freeWAIS-sf	Stable	Private	-
Harvest	www.tardis.ed.ac.uk/harvest	Stable	Private	-
Zebra	Indexdata.dk/zebra	Stable	Private	-
Personal Librarian	www.pls.com	Stable	Private	-